4П

УДК 519.685.4

Потиенко С.В.
Институт кибернетики имени В.М. Глушкова НАН Украины, г. Киев, Украина
stepan@iss.org.ua

Статическая проверка требований и подходы к решению проблемы достижимости

Предложен метод статического анализа систем, представленных в формализме базовых протоколов. Построены алгоритмы проверки таких свойств систем, как непротиворечивость и полнота, а также выполнимость условий целостности. Рассмотрена проблема выявления и достижимости состояний системы, в которых нарушаются заданные свойства. Предложены подходы к решению проблемы достижимости с помощью методов как статического анализа, так и проверки на модели.

Разработка больших программных систем связана с большими трудозатратами как на проектирование, создание программного кода, так и на тестирование и исправление ошибок. При этом исправление ошибок на более поздних стадиях проекта обходится дороже, чем на ранних, таких, как сбор требований и проектирование, а стоимость ошибок, не обнаруженных во время тестирования, может возрасти многократно. Поэтому актуальной проблемой является верификация формальных спецификаций – доказательство свойств полноты, непротиворечивости, целостности и т.д. Логический подход к этой проблеме подразумевает разработку точной математической семантики для языка спецификаций, выбор логического языка представления свойств и выбор соответствующей методики доказательства.

Существует два общеизвестных подхода к верификации – проверка на модели (model checking) и доказательство теорем (theorem proving) [1]. Проверка на модели применяется к моделям с конечным числом состояний и осуществляется как поиск в пространстве состояний. Основная проблема этого подхода лежит в построении и анализе большого пространства состояний, рост которого вызван эффектом комбинаторного взрыва. Доказательство теорем представляет собой технику, в которой и система, и исследуемые свойства выражаются в виде формул в некоторой математической логике. Процесс проверки модели заключается в поиске доказательства выполнимости формул, построенных на основе заданных спецификаций.

В данной работе рассматривается статический анализ систем, представленных в формализме базовых протоколов. Этот подход базируется на статической генерации утверждений в виде некоторых формул с последующими доказательствами их выполнимости, и подразумевает отсутствие известных достижимых состояний системы. Алгоритмы и проблемы доказательства построенных формул оставлены за пределами публикации.

Краткое описание языка базовых протоколов

Базовыми протоколами формализуется набор правил, описывающих поведение анализируемой системы. Базовый протокол представляет собой выражение вида $\forall x(\alpha \to < p > \beta)$, где x – список (типизированных) параметров, α и β – формулы базового логического языка, p – процесс протокола (конечное поведение композиции нескольких агентов и среды, обычно задается с помощью MSC диаграмм). Формула α называется предусловием, а формула β – постусловием базового протокола. Сам

базовый протокол может рассматриваться как формула темпоральной логики, выражающая тот факт, что, если состояние системы имеет разметку, удовлетворяющую условию α, то процесс p может быть инициирован, и после его завершения состояние системы будет удовлетворять условию β. В качестве базового языка используется бескванторный язык многосортного исчисления предикатов.

Базовые протоколы и транзиционные системы

Для статического анализа свойств будем рассматривать размеченную транзиционную систему. Это тройка вида:

$$< S, A, T >,$$

где S – множество состояний транзиционной системы, A – множество меток, T – множество переходов и $T \subseteq S \times A \times S$. Если $s, s' \in S$, $a \in A$, тогда $(s, a, s') \in T$ записывается как:

$$s \xrightarrow{a} s'.$$

Путь в размеченной транзиционной системе – это последовательность переходов $t_1, t_2, \ldots t_i \ldots$, где каждый переход t_i, имеющий метку a_i, переводит систему из состояния s_{i-1} в состояние s_i (s_0 – начальное состояние).

$$s_0 \xrightarrow{a_1} s_1 \xrightarrow{a_2} \ldots \xrightarrow{a_i} \ldots$$

Трасса в транзиционной системе – это последовательность меток $a_1, a_2, \ldots a_i, \ldots$ отвечающих истории $t_1, t_2, \ldots t_i \ldots$.

Рассмотрим систему базовых протоколов как объект статического анализа. Она является спецификацией размеченной транзиционной системы, в которой базовые протоколы играют роль переходов, а их имена – роль меток. В системе базовых протоколов известны только начальные состояния специфицируемой транзиционной системы. Для построения остальных состояний на каждом состоянии из S существует частичное преобразование $\mu : S \to S$.

Вводя некоторые абстракции, вместо конкретных состояний можно оперировать множествами состояний, задаваемыми формулами базового языка. Предусловием α базового протокола задается множество состояний, из которых возможен переход, размеченный именем данного протокола. В этих состояниях базовый протокол называется применимым. Процесс p базового протокола рассматривается как единое целое и не несет функциональной нагрузки. Функцию преобразования одного множества состояний в другое под действием заданного постусловия назовем предикатным трансформером:

$$f' = pt(f, \beta).$$

Здесь f, f' – формулы базового языка, задающие состояния системы до и после применения базового протокола, β – постусловие базового протокола.

Непротиворечивость

Дадим определение недетерминизма. Недетерминированным называется такое состояние s размеченной транзиционной системы, из которого допустим более чем один переход. При моделировании реактивных систем возникает требование проверки детерминированности поведения. Как правило, моделируемая система должна детерминировано реагировать на внешние раздражители.

Дадим определение непротиворечивости для систем базовых протоколов. Назовем два базовых протокола непротиворечивыми, если не существует состояния,

на котором одновременно применимы оба протокола. Это определяется отсутствием пересечений предусловий базовых протоколов:

$$\forall(i,j)\ (\ i \neq j \wedge a_i \in A \wedge a_j \in A \rightarrow \neg(\exists x \alpha_i(x) \wedge \exists y \alpha_j(y))\ ),$$

где $\alpha_i(x), \alpha_j(y)$ – предусловия базовых протоколов a_i и a_j, параметризированных списками x и y.

Утверждение. Непротиворечивая система имеет детерминированное поведение. Действительно, если никакие предусловия не пересекаются, то есть все базовые протоколы непротиворечивы, то в каждом состоянии существует не более одного применимого базового протокола.

Полнота

Помимо детерминизма, критическим требованием к моделируемой системе является отсутствие тупиков.

Дадим определение полноты для систем базовых протоколов. В полной системе должно выполняться следующее условие: дизъюнкция предусловий базовых протоколов приводится к истине:

$$a_1 \in A \wedge a_2 \in A \wedge \ldots \rightarrow \exists x_1 \alpha_1(x_1) \vee \exists x_2 \alpha_2(x_2) \vee \ldots,$$

где $\alpha_i(x_i)$ – предусловие базового протокола a_i, параметризированного списком x_i.

В некоторых реализациях пользователю предоставляется возможность сформулировать ограничения во избежание ложных отрицательных результатов (false negatives). Полнота с учетом пользовательских ограничений определяется так:

$$a_1 \in A \wedge a_2 \in A \wedge \ldots \rightarrow R \vee \exists x_1 \alpha_1(x_1) \vee \exists x_2 \alpha_2(x_2) \vee \ldots,$$

где R – формула базового языка, задающая пользовательские ограничения.

Утверждение. В полной системе нет тупиков. Действительно, если выполняется условие полноты, то в каждом состоянии найдется хотя бы один применимый протокол.

Проверка условий целостности

Кроме полноты и непротиворечивости существует ряд других свойств, обусловленных требованиями к живучести (liveness), надежности, производительности и т.д. моделируемой системы. Такие свойства формулируются для каждой системы индивидуально и представляются в формализме конкретной модели.

Дадим определение условия целостности. Рассмотрим свойство, которое можно записать в виде формулы базового языка. Оно называется условием целостности, если соответствующая формула истинна на всех состояниях системы.

Пусть Q – условие целостности (формула базового языка). Алгоритм проверки поставленного требования «истинности везде» для каждого условия целостности разобьем на три этапа:

Этап 1. Необходимо проверить начальное состояние:

$$s_0 \rightarrow Q,$$

где s_0 – формула базового языка, задающая начальные состояния системы.

Этап 2. Проверим, что не существует базового протокола, который не применим ни в одном состоянии, удовлетворяющем условию целостности Q. То есть предусловия всех базовых протоколов пересекаются с Q:

$$\forall i\ (\ a_i \in A \rightarrow \exists x \alpha_i(x) \wedge Q\ ).$$

Иначе можно сделать два вывода:

– если условие целостности корректно, то, по определению, оно должно выполняться на всех состояниях системы, следовательно, неприменимый на этих состояниях базовый протокол будет недостижим, или

– условие целостности некорректно и должно быть уточнено.

Этап 3. Условие целостности должно быть инвариантно относительно всех базовых протоколов:

$$\forall i\ (\ a_i \in A \rightarrow \forall x(pt(\alpha_i(x) \wedge Q, \beta_i(x)) \rightarrow Q)\ ).$$

Другими словами, если условие целостности выполнялось до применения базового протокола, то оно должно выполняться и после. Для построения формулы, задающей состояния системы после применения базового протокола, используется предикатный трансформер [2], [3].

Анализ результатов статической проверки

Нарушение описанных свойств доказывает существование состояний, в которых система имеет недетерминированное поведение, тупики или нарушаются условия целостности. Назовем такие состояния критическими. В отличие от проверки на модели, описанные методы статического анализа дают полное множество критических состояний, но ничего не говорят об их достижимости. Для проверки достижимости, в первую очередь необходимо построить такие состояния.

Пусть протоколы a_i и a_j противоречивы. По определению, выполняется следующая формула:

$$a_i \in A \wedge a_j \in A \wedge \exists x \alpha_i(x) \wedge \exists y \alpha_j(y).$$

Она определяет критические состояния, в которых система имеет недетерминированное поведение.

Пусть система базовых протоколов неполна с учетом ограничения R. По определению, выполняется следующая формула:

$$a_1 \in A \wedge a_2 \in A \wedge \ldots \wedge \neg(R \vee \exists x_1 \alpha_1(x_1) \vee \exists x_2 \alpha_2(x_2) \vee \ldots\ ).$$

Она определяет критические состояния, являющиеся тупиковыми.

Пусть протокол a_i нарушает условие целостности Q. По определению, существует некоторое выражение базового языка X такое, что выполняется следующая формула:

$$a_i \in A \wedge \exists x(\alpha_i(x) \wedge Q \wedge X \wedge \neg(pt(X, \beta_i(x)) \rightarrow Q)\ ).$$

X определяет критические состояния, в которых нарушается условие целостности Q. Детали построения выражения X здесь не рассматриваются, иначе возникла бы необходимость приводить алгоритмы предикатных трансформеров [2], [3].

Подходы к решению проблемы достижимости

Пусть C – множество критических состояний.

Один из подходов заключается во введении некоторой статической фильтрации. Разобьем множество C на подмножества и/или конкретные состояния произвольным удобным образом. Например, подмножества критических состояний, построенные для каждого найденного нарушения описанных выше свойств, можно рассматривать как удобное разбиение. Каждое подмножество из разбиения задается формулой базового языка, обозначим ее как c_i.

Рассмотрим отрицание формулы c_i как условие целостности: $\neg c_i$. Если система базовых протоколов не нарушает это условие целостности, то состояния, заданные формулой c_i, недостижимы и могут быть отфильтрованы.

С помощью такого подхода можно сделать вывод о недостижимости некоторых критических состояний. Для остальных же (не отфильтрованных) проблема достижимости остается актуальной.

Другой предлагаемый подход основан на использовании методов проверки на модели (model checking). Как упоминалось выше, с помощью этих методов можно доказать достижимость заданных состояний, однако их возможности ограничены из-за наличия проблемы комбинаторного взрыва.

Детальнее рассмотрим анализируемую систему базовых протоколов. Спецификация систем базовых протоколов подробно описана в [4]. Кратко, система состоит из среды и взаимодействующих между собой и с внешним миром агентов. Среда и агенты имеют атрибуты, значениями которых характеризуется состояние системы. У каждого агента существует специальный атрибут *state* символьного типа. Базовые протоколы описывают поведение агентов и определяются следующими правилами:

– пред- и постусловие любого базового протокола содержит выражение, задающее значение атрибута *state* одного и только одного агента, такой агент называется ключевым;
– имя ключевого агента может быть параметризировано;
– так же пред- и постусловие могут содержать формулы базового языка над атрибутами среды и любых агентов.

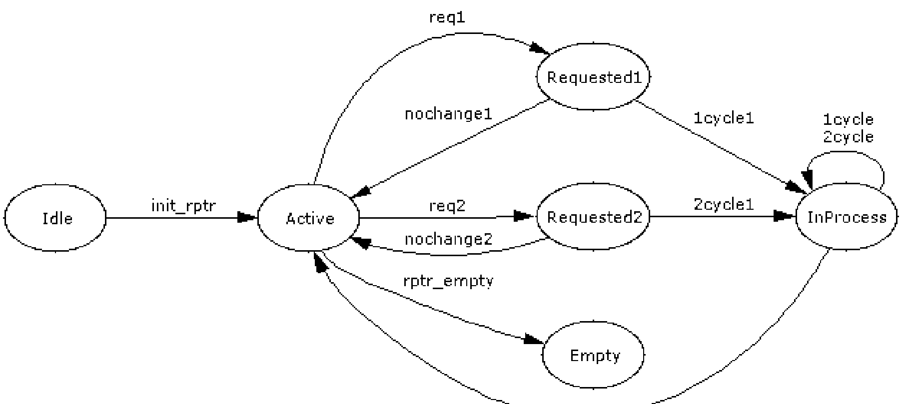
Построим абстракцию анализируемой системы путем элиминации подмножества атрибутов и упрощения пред- и постусловий базовых протоколов. Выберем подмножество, содержащее только специальные атрибуты *state* каждого агента, так как они обладают следующими свойствами в любой системе базовых протоколов:

– присутствуют в предусловии каждого протокола;
– всегда имеют конкретное значение;
– не присутствуют в выражениях с другими атрибутами.

Каждый абстрактный базовый протокол задает изменение атрибута *state* одного агента (другие атрибуты отсутствуют). Таким образом для каждого агента *z* можно построить такой ориентированный граф переходов, что:

– вершины графа ui определяют значения атрибута state: statez = ui, i = 1,…n, где n – кол-во внутренних состояний агента;
– ребра графа aj определяют имена соответствующих базовых протоколов, j = 1,…m, где m – кол-во базовых протоколов, у которых агент z является ключевым.

Приведенная диаграмма иллюстрирует пример графа переходов агента абстрактной системы.



Здесь *Idle, Active, Requested1, Requested2, Empty, InProcess* – внутренние состояния данного агента; *init_rptr, req1, nochange1, req2, nochange2, rptr_empty, ready, 1cycle1, 2cycle1, 1cycle, 2cycle* – базовые протоколы.

Рассмотрим критическое состояние *c* и найдем такие вершины u_i, что:

$$(state_z = u_i) \wedge c \neq \varnothing .$$

Построим все пути из вершины, характеризующей начальное состояние агента в найденные вершины. Такое множество путей можно использовать для проверки достижимости критического состояния *c* с помощью направленного поиска в проверке на модели [5].

Выводы

В данной работе приведены методы статического анализа систем, представленных в формализме базовых протоколов. Построены алгоритмы проверки таких свойств систем, как непротиворечивость и полнота, а также выполнимость условий целостности. Рассмотрена проблема выявления и достижимости состояний системы, в которых нарушаются заданные свойства. Предложены подходы к решению проблемы достижимости с помощью методов как статического анализа, так и проверки на модели.

Литература

1. Clarke E.M. and Wing J.M. ormal methods: State of the art and future directions // Tech. Rep. CMU-CS-96-178. – Carnegie Mellon University (CMU). –1996.
2. Летичевский А.А. (мл.) Об одном классе базовых протоколов // Проблемы программирования. – 2005. – № 4.
3. Летичевський О.О., Левченко І.А. Символьна трасова генерація // Тези доп. Міжнар. конф. «Теоретичні та прикладні аспекти побудови програмних систем TAAPSD'2005». – Київ: НаУКМА, Національний ун-т ім. Т.Г. Шевченка, Ін-т програмних систем НАН України. – 2005.
4. Летичевский А.А., Капитонова Ю.В., Волков В.А., Летичевский А.А. (мл.), Баранов С.Н., Котляров В.П., Вейгарт Т. Спецификация систем с помощью базовых протоколов // Кибернетика и системный анализ. – 2005. – 4.
5. Колчин А.В. Направленный поиск в верификации формальных моделей // Тези доп. Міжнар. конф. «Теоретичні та прикладні аспекти побудови програмних систем TAAPSD'2007». – Бердянськ: НаУКМА, національний ун-т ім. Т.Г. Шевченка, Ін-т програмних систем НАН України. – 2007. – С. 256-258.
6. Letichevsky and Gilbert D. A Model for Interaction of Agents and Environments // Bert D., Choppy C., Moses P. Recent Trends in Algebraic Development Techniques. Lecture Notes in Computer Science 1827, Springer. – 1999.
7. Letichevsky A., Kapitonova J., Letichevsky A.Jr., Volkov V., Baranov S., Kotlyarov V., Weigert T. Basic Protocols, Message Sequence Charts, and the Verification of Requirements Specifications // Proc. International Workshop, WITUL’04. – Rennes (France). – 2004.

S.V. Potyenko

Static Requirements Checking and Approaches to Reachability Problem

A method of static analysis is suggested for the systems represented in a form of basic protocols. Algorithms of checking properties of the system are built for consistency and completeness, satisfiability of safety conditions. The problem of search and reachability of states of the system where given properties broken is considered. Approaches to reachability problem solution are suggested using as static analysis as model checking methods.

Статья поступила в редакцию 18.07.2008.